APPENDIX A

fs_purge.script

```
isql -Usa -Pjc4251 -e <<!
use faultdb1
go

drop proc fs_purgen
go

/************************************
** Create the PURGE stored procedure
************************************/

CREATE procedure fs_purgen
(
        @db_name varchar(30),
        @segment_name varchar(30),
        @space_left int,
        @status int
)
as

/*****************************************************************
** purge procedure will now run every hour to avoid LONG
** lockups at midnight. Just in case we have an extremely busy
** hour, limit row count to 45k (15 traps/sec) to avoid running out
** of locks
*****************************************************************/
        set rowcount 1500000
        DECLARE @activeAlarmRetainDays int
        DECLARE @closedAlarmRetainDays int
        DECLARE @dateStamp varchar(40)

        SELECT @activeAlarmRetainDays = (SELECT ActiveAlarmRetainDays
                FROM FaultServerConfig)

        SELECT @closedAlarmRetainDays = (SELECT ClosedAlarmRetainDays
                FROM FaultServerConfig)

        SELECT @dateStamp = (getdate())

        SET transaction isolation level 0
        PRINT "Purge started ...             (%1!)", @dateStamp
        PRINT "Purging old opentraps ...     (%1!)", @dateStamp

        BEGIN tran
                DECLARE @CustomerID binary(4)
                SELECT @CustomerID = 0

                DECLARE openPurgeAlarmKeyCur cursor FOR
                        SELECT AlarmKey FROM OpenAlarm WHERE
                        TimeStamp < dateadd(day, -@activeAlarmRetainDays, getdate())

                OPEN openPurgeAlarmKeyCur

                DECLARE @tempAlarmKey numeric(30, 0)
                FETCH openPurgeAlarmKeyCur into @tempAlarmKey
                WHILE (@@sqlstatus = 0)
                BEGIN
                        UPDATE Alarm
                        SET ActiveBit = 0
                        WHERE AlarmKey = @tempAlarmKey
                        DECLARE @RowKey_bin binary(4)
                        SELECT @RowKey_bin = convert(binary(4),convert(int,@tempAlarmKey))
                        EXEC DBAction_insert 'OpenAlarm', 'D', @RowKey_bin, @CustomerID

                        FETCH openPurgeAlarmKeyCur into @tempAlarmKey
                END

                CLOSE openPurgeAlarmKeyCur
                DEALLOCATE cursor openPurgeAlarmKeyCur
                delete OpenTrap where TrapKey not in (select TrapKey from OpenEvent)
        COMMIT tran

        SET transaction isolation level 1

/*************************************************************************
** delete closed alarms that are older than the closedAlarmRetainDays and
** the corresponding  Events, GroupAlarms and Traps
*************************************************************************/

        SELECT @dateStamp = (getdate())

        PRINT "Purging old closed alarms ...          (%1!)", @dateStamp

        BEGIN tran
                DELETE Alarm from Alarm a WHERE
                        TimeStamp < dateadd(day, -@closedAlarmRetainDays, getdate()) AND
                        ActiveBit = 0
        COMMIT tran

/*****************************************************
** delete all corresponding Events,GroupAlarms and Traps
*****************************************************/

        SELECT @dateStamp = (getdate())
        PRINT "Purging old group alarms ...           (%1!)", @dateStamp

        BEGIN tran
                DELETE GroupAlarm WHERE
                        GroupAlarmKey not in (SELECT AlarmKey FROM Alarm)
        COMMIT tran

        SELECT @dateStamp = (getdate())
        PRINT "Purging old events ...                 (%1!)", @dateStamp

        BEGIN tran
                DELETE Event FROM Trap t, Event e WHERE
                        e.TrapKey = t.TrapKey AND
                        t.TimeStamp < dateadd(day, -@closedAlarmRetainDays, getdate()) AND
                        e.EventKey not in (SELECT EventKey FROM Alarm) AND
                        e.EventKey not in (SELECT EventKey FROM GroupAlarm)
        COMMIT tran

        SELECT @dateStamp = (getdate())
        PRINT "Purging old traps ...                    (%1!)", @dateStamp

        BEGIN tran
                DELETE Trap WHERE TrapKey not in (SELECT TrapKey FROM Event) AND
                        TimeStamp < dateadd(day, -@closedAlarmRetainDays, getdate())
        COMMIT tran

        SELECT @dateStamp = (getdate())
        PRINT "Purging old CircuitAlarms ...            (%1!)", @dateStamp

        DELETE CircuitAlarm WHERE datediff(day,TimeStamp,getdate())
                > (select  MaxCircuitAlarmDays from FaultServerConfig)
/***********************************************************************
**      MaxCircuitAlarmDays is used for SLA reports
**              > @closedAlarmRetainDays
***********************************************************************/
        SELECT @dateStamp = (getdate())
        PRINT "Purge completed ...                      (%1!)", @dateStamp
go
!
```

APPENDIX B

fs_inserts.script

```
isql -Usa -Pjc4251 -e <<!
use faultdb1
go

drop proc fs_inserts
go

/**********************************
** Create the STATS stored procedure
**********************************/

CREATE procedure fs_inserts
(
        @db_name varchar(30),
        @segment_name varchar(30),
        @space_left int,
        @status int
)
as

/***********************************************************
** count the number of traps and alarms inserted during the last
** hour.
***********************************************************/
        DECLARE @alarmRows int
        DECLARE @trapRows int
        DECLARE @timeStamp varchar(40)
        DECLARE @dateStamp varchar(40)

        SELECT @timeStamp = (dateadd(hour, -1, getdate()))
        SELECT @dateStamp = (getdate())

        SELECT @alarmRows = (SELECT count(*) from Alarm where
                TimeStamp > @timeStamp)

        SELECT @trapRows = (SELECT count(*) from Trap where
                TimeStamp > @timeStamp)

        PRINT "Insert %1!       Alarms since %2! (%3!)",
                @alarmRows, @timeStamp, @dateStamp
        PRINT "Insert %1!       Traps  since %2! (%3!)",
                @trapRows, @timeStamp, @dateStamp

go
!
```

10082864.022602

APPENDIX C

fs_stats.script

```
isql -Usa -Pjc4251 -e <<!
use faultdb1
go

drop proc fs_stats
go

/************************************
** Create the STATS stored procedure
************************************/

CREATE procedure fs_stats
(
        @db_name varchar(30),
        @segment_name varchar(30),
        @space_left int,
        @status int
)
as

/*****************************************************************
** purge procedure will now run every hour to avoid LONG
** lockups at midnight. Just in case we have an extremely busy
** hour, limit row count to 45k (15 traps/sec) to avoid running out
** of locks
*****************************************************************/
        DECLARE @alarmRows int
        DECLARE @trapRows int
        DECLARE @timeStamp varchar(40)
        DECLARE @dateStamp varchar(40)

        SELECT @timeStamp = (dateadd(day, -1, getdate()))
        SELECT @dateStamp = (getdate())

        SELECT @alarmRows = (SELECT count(*) from Alarm where
                TimeStamp < dateadd(day, -1, getdate()))

        SELECT @trapRows = (SELECT count(*) from Trap where
                TimeStamp < dateadd(day, -1, getdate()))

        PRINT "%1!     Alarms older than %2!      (%3!)",
                @alarmRows, @timeStamp, @dateStamp
        PRINT "%1!     Traps  older than %2!      (%3!)",
                @trapRows, @timeStamp, @dateStamp

        SELECT @alarmRows = (SELECT count(*) from Alarm)

        SELECT @trapRows = (SELECT count(*) from Trap)

        PRINT "%1!     Total Alarms                       (%2!)", @alarmRows, @dateStamp
        PRINT "%1!     Total Traps                        (%2!)", @trapRows, @dateStamp
go
!
```

APPENDIX D

fs_stats_hr.script

```
isql -Usa -Pjc4251 -e <<!
use faultdb1
go

drop proc fs_stats_hr
go

/***********************************
** Create the STATS stored procedure
***********************************/
CREATE procedure fs_stats_hr
(
        @db_name varchar(30),
        @segment_name varchar(30),
        @space_left int,
        @status int
)
as
/*****************************************************************
** purge procedure will now run every hour to avoid LONG lockups at midnight. Just in case we have an
** extremely busy hour, limit row count to 45k (15 traps/sec) to avoid running out of locks
*****************************************************************/
        DECLARE @alarmRows int
        DECLARE @trapRows int
        DECLARE @timeStamp varchar(40)
        DECLARE @dateStamp varchar(40)
        DECLARE @activeAlarmRetainDays int
    DECLARE @closedAlarmRetainDays int

        SELECT @dateStamp = (getdate())

    SELECT @activeAlarmRetainDays = (SELECT ActiveAlarmRetainDays
        FROM FaultServerConfig)

    SELECT @closedAlarmRetainDays = (SELECT ClosedAlarmRetainDays
        FROM FaultServerConfig)

        SELECT @timeStamp = (dateadd(day, -@activeAlarmRetainDays, getdate()))

        PRINT "activeAlarmRetainDays %1!", @activeAlarmRetainDays
        PRINT "closedAlarmRetainDays %1!", @closedAlarmRetainDays

        SELECT @alarmRows = (SELECT count(*) from Alarm where
                TimeStamp < dateadd(day, -@closedAlarmRetainDays, getdate()))

        SELECT @trapRows = (SELECT count(*) from Trap where
                TimeStamp < dateadd(day, -@closedAlarmRetainDays, getdate()))

        PRINT "Delete %1!       Alarms before %2! (%3!)",
                @alarmRows, @timeStamp, @dateStamp
        PRINT "Delete %1!       Traps before %2! (%3!)",
                @trapRows, @timeStamp, @dateStamp
go
!
```

APPENDIX E

fault_cron

```
#!/bin/ksh

Uname=`uname -n`

reset_all ()
{
        export cnt_TrapForw cnt_RuleHand cnt_Reliable cnt_DBIdHand cnt_fstrapd
        export cnt_Notifica cnt_AlarmHan cnt_ControlI cnt_AlarmFor cnt_EventHan

        cnt_TrapForw=0
        cnt_RuleHand=0
        cnt_Reliable=0
        cnt_DBIdHand=0
        cnt_fstrapd=0
        cnt_Notifica=0
        cnt_AlarmHan=0
        cnt_ControlI=0
        cnt_AlarmFor=0
        cnt_EventHan=0
}

start_it ()
{

        cnt=0
        export cnt

        Date=`date +'%b %e %T'`

        /etc/rc3.d/S99jws stop
        echo "$Date $Uname fault_cron: /etc/rc3.d/S99jws stop" \
                >>/var/adm/messages

        while true
        do
                if ps -fe | grep -i web | grep -v grep
                then
                        cnt=`expr $cnt + 1`

                        if test $cnt -gt 5
                        then
                                for x in `ps -fe | grep -i web | \
                                        grep -v grep | awk '{print $2}'`
                                do
                                        kill $x
                                done
                                break
                        fi
                        sleep 2
                        continue
                fi
                break
        done

        /etc/rc3.d/S77fsd stop
        sleep 2
        cd /opt/RelyENT/bin/
        /opt/RelyENT/bin/StopAgt.sh

        echo "$Date $Uname fault_cron: /etc/rc3.d/S77fsd stop" \
                >>/var/adm/messages

        while true
        do
                if ps -fe | grep -i cvFault | grep -v grep
                then
                        cnt=`expr $cnt + 1`

                        if test $cnt -gt 5
                        then
                                for x in `ps -fe | grep -i cvFault | \
                                        grep -v grep | awk '{print $2}'`
                                do
                                        Process=`ps -e|grep " $x "|awk '{print $4}'`
                                        echo "$Date $Uname fault_cron: killing $Process" \
                                                >>/var/adm/messages
                                        kill -9 $x
                                done
                                break
                        fi
                        sleep 2
                        continue
                fi
                break
        done

        /etc/rc3.d/S77fsd start
        echo "$Date $Uname fault_cron: /etc/rc3.d/S77fsd start" \
                >>/var/adm/messages

        sleep 4

        /etc/rc3.d/S99jws start
        sleep 2
        /opt/RelyENT/bin/StartAgt.sh
        echo "$Date $Uname fault_cron: /etc/rc3.d/S99jws start" \
                >>/var/adm/messages

        reset_all
        sleep 2
}

String="RuleHan|DBIdHan|EventHa|AlarmHa|TrapForw|Reliabl|fstra|RuleHa|Notific|Contro|AlarmF"

Date=`date +'%b %e %T'`
echo "$Date $Uname fault_cron: Starting" >>/var/adm/messages
reset_all

while true
do

        ps -e|egrep "$String" | awk '{print $4}' >/tmp/res.$$
        Date=`date +'%b %e %T'`
```

```
##for x in TrapForw RuleHand Reliable DBIdHand fstrapd Notifica \
##for x in TrapForw RuleHand Reliable DBIdHand fstrapd Notifica \
        ##AlarmHan ControlI AlarmFor EventHan

for x in TrapForw RuleHand DBIdHand fstrapd Notifica \
        AlarmHan AlarmFor EventHan
do
        if grep $x /tmp/res.$$ >/dev/null
        then
                :
        else
                case $x in
                        TrapForw) cnt_TrapForw=`expr $cnt_TrapForw + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_TrapForw" \
                                        >>/var/adm/messages ;;
                        RuleHand) cnt_RuleHand=`expr $cnt_RuleHand + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_RuleHand" \
                                        >>/var/adm/messages ;;
                        Reliable) cnt_Reliable=`expr $cnt_Reliable + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_Reliable" \
                                        >>/var/adm/messages ;;
                        DBIdHand) cnt_DBIdHand=`expr $cnt_DBIdHand + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_DBIdHand" \
                                        >>/var/adm/messages ;;
                        fstrapd) cnt_fstrapd=`expr $cnt_fstrapd + 1`  ;
                                echo "$Date $Uname fault_cron: $x stopped $cnt_fstrapd" \
                                        >>/var/adm/messages ;;
                        Notifica) cnt_Notifica=`expr $cnt_Notifica + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_Notifica" \
                                        >>/var/adm/messages ;;
                        AlarmHan) cnt_AlarmHan=`expr $cnt_AlarmHan + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_AlarmHan" \
                                        >>/var/adm/messages ;;
                        ControlI) cnt_ControlI=`expr $cnt_ControlI + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_ControlI" \
                                        >>/var/adm/messages ;;
                        AlarmFor) cnt_AlarmFor=`expr $cnt_AlarmFor + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_AlarmFor" \
                                        >>/var/adm/messages ;;
                        EventHan) cnt_EventHan=`expr $cnt_EventHan + 1`
                                echo "$Date $Uname fault_cron: $x stopped $cnt_EventHan" \
                                        >>/var/adm/messages ;;

                esac
        fi
done

if test $cnt_TrapForw -ge 10
then
        start_it
elif test $cnt_RuleHand -ge 10
then
        start_it
elif test $cnt_Reliable -ge 10
then
        start_it
elif test $cnt_DBIdHand -ge 10
then
        start_it
elif test $cnt_fstrapd -ge 10
```

```
        then
                start_it
        elif test $cnt_Notifica -ge 10
        then
                start_it
        elif test $cnt_AlarmHan -ge 10
        then
                start_it
        elif test $cnt_ControlI -ge 10
        then
                start_it
        elif test $cnt_AlarmFor -ge 10
        then
                start_it
        elif test $cnt_EventHan -ge 10
        then
                start_it
        else
                sleep 2
        fi
done

rm -f /tmp/res.$$
```

APPENDIX F

check_inserts.sh

```
Uname=`uname -n`
Prog=`basename $0`
PURGELOG=/opt/cvFaultServer/log/fsPurge.log
CHECKLOG=/opt/cvFaultServer/log/check_insert.log
STARS="**************************"

Date=`date '+%b %e %Y %l:%p'|sed 's/  / /g'`

Alarmcnt=`grep "^Insert " $PURGELOG | tail -2 | grep Alarm | wc -l`
nbrAlarms=`grep "^Insert " $PURGELOG | \
        tail -2 | grep Alarm | awk '{print $2}'`

Alarmdate=`grep "^Insert " $PURGELOG | \
        tail -2 | grep Alarm | awk '{print $9,$10,$11,$12}' | \
        sed 's/[()]//g' | sed 's/:../:/'`

Trapcnt=`grep "^Insert " $PURGELOG | tail -2 | grep Trap | wc -l`
nbrTraps=`grep "^Insert " $PURGELOG | \
        tail -2 | grep Trap | awk '{print $2}'`

Trapdate=`grep "^Insert " $PURGELOG | \
        tail -2 | grep Trap | awk '{print $9,$10,$11,$12}' | \
        sed 's/[()]//g' | sed 's/:../:/'`

if test $Alarmcnt != 1 -a $Trapcnt != 1
then
        echo "fs_inserts not running" >>$CHECKLOG
        exit 1
fi

if test "$Alarmdate" != "$Date" -a "$Trapdate" != "$Date"
then
        echo "date mismatch" >>$CHECKLOG
        exit 1
fi

if test $nbrAlarms -eq 0 -a $nbrTraps -eq 0
then
        echo "`date +'%b %e %T'` $Uname $Prog: Alarms and Traps have stopped" \
                >>$PURGELOG
        echo "$STARS$STARS$STARS" >>$PURGELOG

        echo "`date +'%b %e %T'` $Uname $Prog: Alarms and Traps have stopped" \
                >>$CHECKLOG
        echo "$STARS$STARS$STARS" >>$CHECKLOG

        /usr/local/bin/rmcore

        fault_cron=`ps -fe | grep fault_cron | egrep -v "grep" | \
                awk '{print $2}'`

        if test X"$fault_cron" != X
        then
                echo "`date +'%b %e %T'` $Uname $Prog: kill fault_cron\t$fault_cron" \
                        >>$CHECKLOG
        kill -9 $fault_cron
fi

cnt=0
while true
do
        if test $cnt -gt 2
        then
                break
        fi

        fault_cron=`ps -fe | grep fault_cron | egrep -v "grep" | \
                awk '{print $2}'`

        if test X"$fault_cron" != X
        then
                echo "`date +'%b %e %T'` $Uname $Prog: kill fault_cron\t$fault_cron" \
                        >>$CHECKLOG
                kill -9 $fault_cron
                break
        fi
        cnt=`expr $cnt + 1`
        sleep 4
done

cnt=0
export cnt

Date=`date +'%b %e %T'`

echo "$Date $Uname $Prog: /etc/rc3.d/S99jws stop" >>$CHECKLOG
/etc/rc3.d/S99jws stop >>$CHECKLOG 2>&1

while true
do
        if ps -fe | grep -i web | grep -v grep
        then
                cnt=`expr $cnt + 1`

                if test $cnt -gt 5
                then
                        for x in `ps -fe | grep -i web | \
                                grep -v grep | awk '{print $2}'`
                        do
                                if test X"$x" != X
                                then
                                        Process=`ps -e | grep " $x " | awk '{print $4}'`
                                echo "`date +'%b %e %T'` $Uname $Prog: kill $Process\t$x" \
                                        >>$CHECKLOG
                                        kill -9 $x
                                fi
                        done
                        break
                fi
                sleep 5
                continue
        fi
        break
done
```

```
        echo "$Date $Uname $Prog: /etc/rc3.d/S77fsd stop" \
                >>$CHECKLOG
        /etc/rc3.d/S77fsd stop >>$CHECKLOG 2>&1

        while true
        do
                if ps -fe | grep -i cvFault | grep -v grep
                then
                        cnt=`expr $cnt + 1`

                        if test $cnt -gt 5
                        then
                                for x in `ps -fe | grep -i cvFault | \
                                        egrep -v "fsPurge|check_insert|grep" | awk '{print $2}'`
                                do
                                        if test X"$x" != X
                                        then
                                                Process=`ps -e | grep " $x " | awk '{print $4}'`
                                        echo "`date +'%b %e %T'` $Uname $Prog: kill $Process\t$x" \
                                                >>$CHECKLOG
                                                kill -9 $x
                                        fi
                                done
                                break
                        fi
                        sleep 5
                        continue
                fi
                break
        done

        echo "$Date $Uname $Prog: /etc/rc3.d/S77fsd start" \
                >>$CHECKLOG
        /etc/rc3.d/S77fsd start  >>$CHECKLOG 2>&1

        sleep 4

        echo "$Date $Uname $Prog: /etc/rc3.d/S99jws start" \
                >>$CHECKLOG
        /etc/rc3.d/S99jws start >>$CHECKLOG 2>&1

        echo "$Date $Uname $Prog: Starting fault_cron" \
                >>$CHECKLOG
        /usr/local/bin/fault_cron &

        echo "$Date $Uname $Prog: Restart complete" \
                >>$CHECKLOG
        echo "$STARS$STARS************" \
                >>$CHECKLOG
else
        echo "Okydoky" >>$PURGELOG
        echo "Okydoky" >>$CHECKLOG
fi
```

APPENDIX G

fspurge.sh

```
#!/bin/sh
# Filename: fsPurge.sh
# Description:
#      script to invoke a sybase store procedure to purge the faultdb.
#####################################################################################

USAGE='Usage: fsPurge.sh  server db_name username password sybasePath'
Uname=`uname -n`
Prog=`basename $0`

##################################
# check command line arguments
##################################
if [ $# -ne 5 ]
then
      echo $USAGE
      exit 1
fi

#########################################
# log all outputs from isql statements
#########################################
exec >>/opt/cvFaultServer/log/fsPurge.log 2>&1

SYBASE=$5
export SYBASE
DSQUERY=$1
export DSQUERY
BIN=/opt/cvFaultServer/bin
LOCAL=/usr/local/bin
LOG=/opt/cvFaultServer/log

echo "\n`date +'%b %e %T'` $Uname $Prog: Started"
echo "**********************************************\n"

##############################################################
# check and make sure fsPurge,sh is not already running !
##############################################################
ps -e | grep fsPurge.
cnt=`ps -e | grep fsPurge. | wc -l`

if test $cnt -ge 2
then
      echo "`date +'%b %e %T'` $Uname $Prog: already running"
      exit 1
fi

##############################################################################
# The grep will remove output lines that contain a number surrounded by
# any amount of whitespace. If the number is bounded by words, then it
# will not be removed from the output stream.
##############################################################################

Status=0; export Status
Count=0
```

```
##################################################################
# test for successful conclusion, i.e. no deadlock, for up to
# 10 times (10 minutes) !
##################################################################

$5/bin/isql -U$3 -P$4 2>&1 <<! | egrep -v -f $BIN/exclude

use $2
go

exec fs_stats_hr $2, "", 0, 0
go
!

while true
do

echo "\n`date +'%b %e %T'` $Uname $Prog:"
echo "************************************"

$5/bin/isql -U$3 -P$4 2>&1 <<! | egrep -v -f $BIN/exclude

use $2
go

exec fs_purgen $2, "", 0, 0
go

!
        Status=`tail /opt/cvFaultServer/log/fsPurge.log | \
                grep "^(return status" | tail -1`

        #######################
        # did purge succeed ??
        #######################
        if test X"$Status" = X"(return status = 0)"
        then
                sleep 5
                break
        fi

        Count=`expr $Count + 1`

        #########################
        # try up to 10 more times
        #########################
        if test $Count -ge 10
        then
                echo "Purge failed at `date +'%b %e %T'`"
                sleep 5
                break
        fi

        sleep 60
done

echo "\n`date +'%b %e %T'` $Uname $Prog"
echo "************************************"
```

```
$5/bin/isql -U$3 -P$4 <<! | egrep -v -f $BIN/exclude

sp_helpdb  $2
go
!

echo "\n`date +'%b %e %T'` $Uname $Prog"
echo "****************************************"

$5/bin/isql -U$3 -P$4 <<! | egrep -v -f $BIN/exclude

use $2
go

exec fs_inserts $2, "", 0, 0
go
!

nohup $LOCAL/check_inserts.sh >>/opt/cvFaultServer/log/check_insert.log 2>&1 &

echo "\n`date +'%b %e %T'` $Uname $Prog: Completed"
echo "**************************************************"

if test -f $LOG/DBId*
then
        df -k /opt
        rm -f $LOG/*Buffer*
        df -k /opt
        echo
fi
```